

**EV316936415**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

**APPLICATION FOR LETTERS PATENT**

**Quick Starting Video Content**

**Inventors:**

Sumedh N. Barde

Sohail Baig Mohammed

ATTORNEY'S DOCKET NO. MS1-1543US

1     **TECHNICAL FIELD**

2             The present disclosure generally relates to streaming media, and more  
3 particularly, to achieving a quick startup and seamless playing of streaming video  
4 content.

5  
6     **BACKGROUND**

7             Content streaming, such as the streaming of audio, video, and/or text media  
8 content is becoming increasingly popular. The term “streaming” is typically used  
9 to indicate that the data representing the media is provided over a network to a  
10 client computer and the client computer renders the streaming content as it is  
11 received from a network server, rather than waiting for an entire “file” to be  
12 delivered.

13            When media content is “streamed” over a network, it is typically streamed  
14 in data packets. However, there is typically no guarantee that the data packets will  
15 arrive at their destination in the same order in which they are sent, or even that  
16 they will arrive at their destination at all. Additionally, there is typically no  
17 guarantee that the time it takes a data packet to travel from the source to the  
18 destination will be of specific duration, or even that the time will be the same for  
19 different data packets.

20            In order to account for these variances in data delivery to a client computer,  
21 the client computer maintains a buffer of data, typically on the order of three to  
22 five seconds worth of data. This buffer allows the client computer to smooth out  
23 the variances in data delivery so that they are not as noticeable to the user during  
24 playback of the content. However, one problem with this buffering is that it causes  
25 the user to have to wait for the buffer to be initially filled before playback of the

1 content begins. Thus, upon selecting content for playback, users experience a  
2 delay in the start of playback that is generally about as long as the amount of time  
3 set in the buffer on the playback application (e.g., 3 - 5 seconds). This delay can  
4 be longer depending on the buffer setting and network glitches affecting the data  
5 transfer rate.

6 As an example, when a user selects content for playback through a media  
7 playback application running on a client computer, a blank screen is typically  
8 displayed along with a buffering message indicating that a certain percentage of  
9 the content is buffered. The blank screen and the buffering message continue to be  
10 displayed until the percentage of buffering reaches 100%, at which time the  
11 content begins playing through the playback application's user interface. Such  
12 buffering delays are generally regarded as an unavoidable cost of playing  
13 streaming media content (e.g., video content) over the Internet. However, users  
14 who are accustomed to the instant startup behavior of television can find buffering  
15 delays to be quite annoying. Thus, buffering delay has been one of the hindrances  
16 in the mass adoption of current general-purpose streaming technologies.

17 Accordingly, the need exists for ways to reduce the waiting time involved in  
18 playing video content over a network such as the Internet.

## 19 20 **SUMMARY**

21 Quick starting video content is described herein.

22 In accordance with one implementation, a static image is received from a  
23 content provider and displayed while the video content is buffered from the  
24 content provider. The static image can be displayed until the buffering is  
25

1 complete, or, for a minimum fixed duration if the buffering is complete prior to  
2 expiration of the fixed duration.

3 In accordance with another implementation, the last frame of a first video  
4 clip can be displayed while a second video clip continues buffering if the second  
5 clip is not fully buffered by the time the first clip is done playing.

6 In accordance with another implementation, a static image can be buffered  
7 during playback of a first video clip and then displayed while a second video clip  
8 is buffered. The static image can be displayed until the buffering of the second  
9 clip is complete, or for a minimum fixed duration if the buffering happens to be  
10 complete prior to expiration of the fixed duration.

## 11 12 **BRIEF DESCRIPTION OF THE DRAWINGS**

13 The same reference numerals are used throughout the drawings to reference  
14 like components and features.

15 Fig. 1 illustrates an exemplary network environment for implementing  
16 quick starting video content.

17 Fig. 2 illustrates exemplary client and server devices.

18 Fig. 3 illustrates an example graphical user interface of a streaming media  
19 player being implemented through an Active X control in an HTML window.

20 Fig. 4 illustrates an example of a play-list in the prior art that might  
21 represent a play-list accessed when a user selects video content for playback.

22 Fig. 5 illustrates an example of a time line for buffering and playing back  
23 video content.

24 Fig. 6 illustrates an example graphical user interface of a streaming media  
25 player being implemented through an Active X control in an HTML window.

1 Fig. 7 illustrates an example of a time line for buffering and playback of  
2 video content.

3 Fig. 8 illustrates an embodiment of a stitched-reference play-list configured  
4 to control a streaming media player.

5 Fig. 9 provides an example of a still image logo displayed on an example  
6 graphical user interface of a streaming media player.

7 Fig. 10 illustrates another exemplary embodiment of a stitched-reference  
8 play-list configured to control a streaming media player.

9 Fig. 11 illustrates another exemplary embodiment of a stitched-reference  
10 play-list configured to control a streaming media player.

11 Fig. 12 illustrates another exemplary embodiment of a stitched-reference  
12 play-list configured to control a streaming media player.

13 Figs. 13 - 17 illustrate block diagrams of exemplary methods for achieving  
14 quick startup of video content in a streaming media player.

15 Fig. 18 illustrates an exemplary computing environment suitable for  
16 implementing a client computing device and a content server computing device.

## 17 **DETAILED DESCRIPTION**

### 18 **Overview**

19 The following discussion is directed to systems and methods for achieving  
20 quick startup of streaming video content. Streaming video content is provided in  
21 two parts generally comprising a static/still image and dynamic video content.  
22 “Static image” and “still image” are used synonymously throughout this  
23 description. The static image can be the first frame of the video content that  
24 represents an introductory section of the video content depicting a motionless  
25

1 image (e.g., an FBI warning). The static image may also be a logo or  
2 advertisement inserted by a content provider, when, for example, the video content  
3 does not include an introductory portion depicting a motionless image. A  
4 streaming media player is configured to buffer the static image and display it  
5 promptly while buffering the remaining video content in the background.

6 Benefits of the disclosed systems and methods include a reduced waiting  
7 period between the selection of video content and the playback of content by a  
8 media player. There are both real and perceived reductions in the time between  
9 video content selection and video content playback. The reduction in time  
10 between the selection of media content and the playback of content on a media  
11 player makes the streaming video experience via a network (e.g., the Internet)  
12 closer to the experience of watching television.

### 13 14 **Exemplary Environment**

15 Fig. 1 illustrates an exemplary network environment 100 suitable for  
16 achieving quick startup of streaming video content. In the exemplary network  
17 environment 100, multiple (n) client computing devices 102(1), 102(2), . . . , 102(n)  
18 are coupled to multiple (m) content server computing devices 104(1), 104(2), . . . ,  
19 104(m) via a network 106. Network 106 is intended to represent any of a variety  
20 of conventional network topologies and types (including optical, wired and/or  
21 wireless networks), employing any of a variety of conventional network protocols  
22 (including public and/or proprietary protocols). Network 106 may include, for  
23 example, the Internet as well as possibly at least portions of one or more local area  
24 networks (LANs) and/or wide area networks (WANs).

1 Requests from a client device 102 for streaming media content that is  
2 available on a content server device 104 are routed from the client device 102 to  
3 the server device 104 via network 106. The content server device 104 receives the  
4 request and returns the requested content to the requesting client device 102 via  
5 network 106. One or more proxy servers (not shown) may be part of network 106,  
6 and requests from client device 102 and responses to client device 102 may be sent  
7 to and received from such a proxy server(s) rather than the actual content server  
8 device 104. Whatever device (whether it be a content server, proxy server, or other  
9 device) is streaming media content to a client device 102 may be referred to as the  
10 source device for that streaming media content.

11 Computing devices 102 and 104 can each be any of a variety of  
12 conventional computing devices, including desktop PCs, notebook or portable  
13 computers, workstations, mainframe computers, Internet appliances, gaming  
14 consoles, handheld PCs, cellular telephones or other wireless communications  
15 devices, personal digital assistants (PDAs), combinations thereof, and so on. One  
16 or more of devices 102 and 104 can be the same types of devices, or alternatively  
17 different types of devices. An exemplary computing environment for  
18 implementing a client computing device 102 and a content server computing  
19 device 104 is described in more detail herein below with reference to Fig. 18.

20 Content server devices 104 can make any of a variety of data available for  
21 streaming to clients 102. As used herein, “streaming media”, “streaming video”,  
22 “video content”, “video clip”, and any variation thereof generally includes  
23 audio/video content. The term “streaming” is used to indicate that the data  
24 representing the media content is provided over a network to a client device and  
25 that playback of the content can begin prior to the content being delivered in its

1 entirety. The data may be publicly available or alternatively restricted (e.g.,  
2 restricted to only certain users, available only if the appropriate fee is paid, etc.).  
3 The data may be any of a variety of one or more types of content, such as audio,  
4 video, text, images, animation, etc. Additionally, the data may be "on-demand"  
5 (e.g., pre-recorded and of a known size) or alternatively "broadcast" (e.g., having  
6 no known size, such as a digital representation of a concert being captured as the  
7 concert is performed and made available for streaming shortly after capture).

### 8 9 **Exemplary Embodiments**

10 Fig. 2 illustrates an exemplary embodiment of a client device and a server  
11 device configured to achieve quick startup when playing back streaming video  
12 content on a streaming media player. Client device 102 includes a streaming  
13 media player 200 configured to access a streaming control module 202 of a content  
14 server device 104. As mentioned above, content server device 104 may be any  
15 typical source server device or appropriate alternative device such as, for example,  
16 a proxy server device. Although not shown in Fig. 2, one or more additional  
17 devices (e.g., firewalls, routers, gateways, bridges, multiple proxy servers, etc.)  
18 may be situated between client device 102 and content server device 104. It  
19 should be noted that multiple clients 102 may access content server 104 and that a  
20 single client 102 may access multiple content servers 104, although only a single  
21 client 102 and content server 104 have been shown in Fig. 2 for ease of  
22 explanation.

23 Content server device 104 includes one or more streaming media content  
24 files 204 from which a selection can be made by media player 200 (e.g., based on  
25 user input at player 200) on client device 102. Selections are made through

1 streaming control module 202 and selected content files 204 are streamed to player  
2 200. Client device 102 includes a buffer 206 that is used to buffer streaming  
3 media content received from content server 104. Streaming media player 200 is  
4 configured to buffer and play back streaming media content in accordance with a  
5 stitched-reference play-list 208. Although illustrated as part of streaming media  
6 player 200, both the buffer 206 and stitched-reference play-list 208 may  
7 alternatively be separate from streaming media player 200.

8 Streaming control module 202 typically streams media content to a  
9 streaming media player 200 at the steady state rate of the streaming media content.  
10 Typically, the steady state rate of the streaming media content is the rate at which  
11 the streaming media content is encoded for playback (typically referred to as the  
12 encoded bit rate of the content), although alternatively this rate may vary.  
13 Different content can be encoded at different rates, which generally affect the  
14 quality of the content. Higher encoded bit rates typically result in higher quality  
15 content playback. Additionally, different portions of content can be encoded at  
16 different rates, also referred to as variable bit rate encoding (e.g., quiet periods of a  
17 song may be encoded at 60 kbps (kilobits per second) while more "active" portions  
18 of the song may be encoded at 150 kbps).

19 Communication between devices 102 and 104 can occur using a variety of  
20 different protocols. In one implementation, communication between devices 102  
21 and 104 occurs using a version of the HyperText Transport Protocol (HTTP), such  
22 as version 1.0 (HTTP 1.0) or version 2.0 (HTTP 2.0). In another implementation,  
23 communication between devices 102 and 104 occurs using the Real Time  
24 Streaming Protocol (RTSP), such as RTSPU (RTSP used with User Datagram  
25 Protocol (UDP)) and/or RTSPT (RTSP used with Transmission Control Protocol

1 (TCP)). Alternatively, other protocols may be used, such as the Session Initiation  
2 Protocol (SIP), the Simple Object Access Protocol (SOAP), the Microsoft Media  
3 Server (MMS) protocol (such as MMSU (MMS used with UDP) or MMST (MMS  
4 used with TCP), and so forth.

5 Additionally, streaming media content can be stored and streamed in  
6 accordance with any of a variety of different streaming media formats. In one  
7 exemplary implementation, media is streamed in accordance with the ASF format  
8 (Advanced Systems Format or Advanced Streaming Format). Additional  
9 information regarding ASF is available from Microsoft® Corporation of Redmond,  
10 Washington. Alternatively, or in conjunction with the ASF format, other streaming  
11 media formats may be used such as WMA (Windows Media Audio), WMV  
12 (Windows Media Video), MPEG (Moving Pictures Experts Group)-1, MPEG-2,  
13 MPEG-4, Quicktime, etc.

14 Different pieces of streaming media content are illustrated in Fig. 2 as  
15 different content files 204. In general, a piece of streaming media content may be  
16 stored as multiple files, depicted for example, by content files 204, static image  
17 files 210, and video content files 212. The manner in which a "piece" of content is  
18 defined can vary by implementation and can be based on the type of media. For  
19 example, for video content, each video clip can be a piece of content. Content may  
20 be separated into pieces along natural boundaries (e.g., different video clips), or  
21 alternatively in other arbitrary manners (e.g., every five minutes of content is a  
22 piece).

23 Figs. 3, 4 and 5 generally illustrate an exemplary prior art process for  
24 accessing streaming video content via a streaming media player such as the  
25 streaming media player 200 of Fig. 2. Fig. 3 shows an example graphical user

1 interface of a streaming media player being implemented through an Active X  
2 control in an HTML window. The HTML window of Fig. 3 illustrates an example  
3 of a news-oriented web site offering access to video clips 300 of daily news  
4 events. It should be noted that streaming media players may generally be  
5 implemented in various forms such as a stand-alone application executing on a  
6 client computing device 102, for example, or as an HTML embedded player as  
7 illustrated in Fig. 3.

8 As indicated in Fig. 3, a user may select 302 video content (e.g., Video 2)  
9 by clicking on a video link 300 from an HTML-based Internet web page. A video  
10 link 300 generally points to a play-list that contains one or more references to  
11 video content. Fig. 4 illustrates an example of a play-list 400 in the prior art that  
12 might represent a play-list accessed during the scenario of Fig. 3 in which a user  
13 selects Video 2 for playback. A play-list 400 generally includes one or more  
14 references to various media content along with instructions to control a streaming  
15 media player regarding the access, buffering, and playback of the media content. A  
16 play-list 400 may be stored on any of various server devices accessible to client  
17 device 102 via network 106, but is more typically stored either on the server that  
18 supports the web site providing the link 300 to the media content or on the content  
19 server 104 that stores the media content to which the play-list 400 refers.

20 The example prior art play-list 400 of Fig. 4 is implemented as an ASX  
21 metafile, which is a type of Windows Media metafile used with Windows Media  
22 Player®. The ASX file generally redirects streaming media content away from a  
23 browser to Windows Media Player. Other types of metafiles are available for other  
24 streaming media players, such as RAM or RPM metafiles used for Real System's  
25 RealPlayer. In general, play-lists implemented as ASX metafiles include various

1 ASX syntax elements that provide identification information 402 about the file,  
2 such as the type of file, the author, the title, and so on. ASX metafiles also include  
3 reference information 404 that includes a path to various media content such as  
4 streaming video content. The example play-list 400 of Fig. 4 provides a media  
5 player with the location to streaming video content, such as the content for Video 2  
6 as shown in Fig. 3.

7 A streaming media player implements a play-list 400 and accesses the  
8 referenced content in the order in which it appears in the play-list 400. Although  
9 there is only one reference to content (i.e., to Video2.asf) in the example play-list  
10 400 of Fig. 4, play-lists commonly have a number of references to streaming  
11 content. A streaming media player begins buffering the referenced streaming  
12 video content and begins playing it back as soon as the buffer is completely filled.  
13 A buffer in a media player (e.g., buffer 206 in media player 200) can usually be set  
14 by a user to contain a certain amount of pre-stored content. Buffers are typically  
15 set to store content that will support an amount of playback time on the order of 3  
16 to 5 seconds. Because the steady state rate of streaming media content is typically  
17 the rate at which the streaming media content is encoded for playback, a 5 second  
18 buffer setting, for example, would generally result in a 5 second waiting period  
19 from the time a user makes a video selection to the time the buffer is full and the  
20 video playback begins.

21 Fig. 5 illustrates an example of a time line for buffering and playing back  
22 video content where a media player buffer is set for 5 seconds. This example time  
23 line is applicable to the exemplary prior art process discussed above with respect to  
24 Fig. 3. Thus, at time zero 302 (Fig. 5) a user selects 302 Video 2 (Fig. 3) for  
25 playback. The user then experiences a delay while the video content buffers.

1 During the buffering delay, the user sees a blank media screen 304 and a buffering  
2 message 306 that indicates a certain percentage of the buffering is complete (e.g.,  
3 18% complete). Fig. 5 illustrates that the buffer time has been set to 5 seconds.  
4 The blank media screen 304 is displayed until the buffering is 100% complete,  
5 which typically occurs 5 seconds after the video selection is made, since the steady  
6 state rate of streaming media content is typically the rate at which the streaming  
7 media content is encoded for playback. However, the delay period and display  
8 time of a blank media screen 304 may vary depending on network glitches and  
9 other factors that can affect the data transfer rate and buffering time. The video  
10 selection (i.e., Video 2) begins playing back when the buffer is full.

11 In addition to showing how a user experiences a buffering delay upon  
12 making a video selection, the time line of Fig. 5 also illustrates how the first  
13 portion of the video content to be played back may simply comprise a static/still  
14 image such as, for example, an "FBI WARNING". Thus, after the buffering delay  
15 of 5 seconds, the user may also wait an additional period of time as a still image is  
16 displayed (e.g., 6 seconds as shown in the example of Fig. 5). Assuming the video  
17 is 25 frames per second, for example, the first 6 seconds of the video may require  
18 the buffering of between 1 and 150 frames, depending on how the content was  
19 encoded. These frames are then rendered with no change in the actual display of  
20 the video content. Only after the original still image has buffered, can the  
21 remaining video content begin to buffer. Thus, even though the still video image  
22 displays only a single static image for 6 seconds, it nevertheless requires time to  
23 fully buffer as if it were typical dynamic video content. Depending on how many  
24 frames are being buffered for the still video image, the buffering of the remaining  
25

1 dynamic video content could begin as much as 6 seconds later. These delays are  
2 significant drawbacks to the overall experience of a streaming video experience.

3 Accordingly, an embodiment is described with reference to Figs. 2, 6, 7 and  
4 8 that illustrates quick startup of streaming video content on a streaming media  
5 player 200. Fig. 6 is similar to Fig. 3 above in that it illustrates an example  
6 graphical user interface of a streaming media player 200 being implemented  
7 through an Active X control in an HTML window. The HTML window illustrates  
8 an example of a news-oriented web site offering access to video clips 600 of daily  
9 news events. Fig. 6 is intended to indicate a user selecting (with pointer 602)  
10 video content titled Video 11. Unlike the prior art example of Fig.3, the Fig. 6  
11 embodiment is intended to indicate that an "FBI WARNING" screen 604 is  
12 displayed almost directly (i.e., without delay) after a user selects the video content.  
13 In addition, the buffering message 306 previously displayed in the Fig. 3 example  
14 is not displayed at location 606 in the Fig. 6 embodiment.

15 The timing for buffering and playback of the Video 11 content of Fig. 6 is  
16 illustrated by the video playback timeline of Fig. 7. Fig. 7 shows that Video 11 is  
17 selected at time zero 602. A still image configured, for example, as a "JPEG"  
18 (Joint Photographic Experts Group) file (i.e., the FBI Warning), is then buffered  
19 for a short period of time (e.g., 0.5 seconds). After the FBI Warning JPEG is  
20 buffered, it is displayed for 6 seconds. This is similar to the 6 second display of  
21 the static video image of the FBI Warning discussed above regarding Figs. 3 - 5.  
22 However, the still image in the current embodiment of Figs. 6 and 7 is rendered as  
23 a single image JPEG file, rather than as numerous frames of video content as in the  
24 prior art example of Figs. 3 - 5. Thus, the FBI Warning is displayed in Fig. 6 as a  
25 still image JPEG file until 6.5 seconds on the timeline of Fig. 7. In addition, since

1 the FBI Warning is displayed as a still image JPEG file, the moment it is done  
2 buffering and begins being displayed, the remainder of the video content can begin  
3 buffering in the background. If the buffer time is set to 5 seconds, for example, the  
4 remaining video content will be fully buffered at 5.5 seconds. Thus, if necessary,  
5 the remaining video content could begin playing at 5.5 seconds.

6 Fig. 8 illustrates an embodiment of a stitched-reference play-list 208 (Fig.  
7 2) that is configured (e.g., by a content provider) to control a streaming media  
8 player 200 to access, buffer, and playback the Video 11 content as discussed above  
9 with respect to Figs. 6 and 7. The discussion above regarding a typical play-list  
10 400 is generally applicable to a stitched-reference play-list 208. Thus, a stitched-  
11 reference play-list 208 is configured and implemented in much the same way as a  
12 typical play-list 400. However, as defined and used herein, a stitched-reference  
13 play-list 208 is distinguishable from a typical play-list in one or more ways.

14 For example, a stitched-reference play-list 208 refers to a single play-list  
15 file that includes a reference to a static image and a reference to video content that  
16 are “stitched” together in the play-list file in order to facilitate a quick startup of  
17 streaming video content through a streaming media player 200 that is capable of  
18 supporting the playback of still images in popular formats such as JPEGs (Joint  
19 Photographic Experts Group), GIFs (Graphics Interchange Format), PNGs  
20 (Portable Network Graphics), and the like. Thus, in general, a content provider  
21 can facilitate quick startup of streaming video content by providing the content as  
22 two pieces stitched together in a stitched-reference play-list 208 configured, for  
23 example, as an ASX play-list file. The first piece of content would be a static  
24 image, and the second piece would be the remaining dynamic video content. In  
25 addition, a stitched-reference play-list 208 may also include instructions to control

1 the playback of a static image 210 with respect to the playback of video content  
2 212 that are included in one or more content files 204 on a content server 104 such  
3 as shown in Fig. 2.

4 Accordingly, the Fig. 8 embodiment of a stitched-reference play-list 208  
5 includes two entries 800 and 802; the first entry 800 providing a reference to the  
6 FBIWarning.jpg still image 804 displayed as 604 in Fig. 6, and the second entry  
7 802 providing a reference to the remaining dynamic video content of Video11.asf  
8 806 that has been selected 602 in Fig. 6. Thus, when a user selects video content  
9 600 as shown in Fig. 6, such as Video 11, for example, a link is made to a stitched-  
10 reference play-list 208 such as that shown in Fig. 8. The stitched-reference play-  
11 list 208 may be accessed by a media player 200 from a content server 104 such as  
12 illustrated in Fig. 2, or it may be accessed from any other device on which it might  
13 be stored that is accessible via network 106.

14 A streaming media player 200 implements the stitched-reference play-list  
15 208 in the same manner as it would implement a typical play-list. That is, the  
16 streaming media player 200 accesses the referenced content in the order in which it  
17 appears in the play-list 208. In addition, however, the streaming media player 200  
18 implements commands in the stitched-reference play-list 208 that facilitate the  
19 quick display of referenced content and transitions between the referenced content.  
20 For example, in accordance with the stitched-reference play-list 208 of Fig. 8,  
21 when a user selects Video 11 as shown in Fig. 6, a streaming media player 200 first  
22 accesses and buffers the FBIWarning.jpg still image 804. As shown in Fig. 7, the  
23 buffering of a still image 804 can occur in a very short period of time, such as 0.5  
24 seconds. Thus, after a mere 0.5 seconds from the moment video content is  
25 selected, a user experiences content displayed by the streaming media player 200.

1 The media player 200 continues to implement the stitched-reference play-list 208  
2 of Fig. 8 by executing a “Duration” command 808 and a “ShowWhileBuffering”  
3 command 810. As these commands are executed, the streaming media player 200  
4 buffers the remaining dynamic video content of Video 11 806 that is referenced in  
5 the second entry 802. The duration command 808 is set to a value of 6 seconds,  
6 which means that the media player 200 will display the still image 804 (i.e., the  
7 FBIWarning.jpg) for at least 6 seconds. Because the value of the  
8 ShowWhileBuffering command 810 is set to “true”, the media player 200 will  
9 additionally ensure that the still image 804 continues to be displayed until the  
10 remaining video content of Video11.asf 806 is fully buffered. Thus, the stitched-  
11 reference play-list 208 of Fig. 8 directs the streaming media player 200 to display  
12 the still image 804 (i.e., the FBIWarning.jpg) for as long as it takes the remaining  
13 video content of Video11.asf 806 to be fully buffered, but at least for 6 seconds.

14 Revisiting the timeline of Fig. 7 is helpful in describing the stitched-  
15 reference play-list 208 of Fig. 8. When a user selects Video 11 at time zero 602,  
16 the streaming media player 200 is directed by a link to the stitched-reference play-  
17 list 208 of Fig. 8. The media player 200 implements the play-list 208 by accessing  
18 referenced content in the order in which it appears in the play-list 208. Thus, the  
19 FBIWarning.jpg still image 804 is accessed and buffered. It takes approximately  
20 0.5 seconds to buffer the still image 804. As Fig. 7 indicates, the media player 200  
21 displays the still image 804 after it is buffered. Fig. 7 additionally shows how the  
22 still image 804 remains displayed for 6 full seconds as directed by the duration  
23 command 808 in the stitched-reference play-list 208 of Fig. 8. During the 6  
24 seconds, since the still image 804 is already fully buffered and being displayed, the  
25 streaming media player 200 can buffer the remaining video content of Video11.asf

1 806. Since the buffer in the media player 200 is set to 5 seconds, the remaining  
2 video content of Video11.asf 806 is usually fully buffered within 5 seconds from  
3 the moment that the still image 804 begins being displayed (i.e., at 0.5 seconds).  
4 Thus, the remaining video content of Video11.asf 806 is fully buffered at 5.5  
5 seconds (Fig. 7). Therefore, the remaining video content of Video11.asf 806 could  
6 begin playing back starting at 5.5 seconds. However, the duration command 808  
7 in the stitched-reference play-list 208 of Fig. 8 ensures that the FBIWarning.jpg  
8 still image 804 is displayed for at least a full 6 seconds. Therefore, the remaining  
9 video content of Video11.asf 806 does not begin playing back until 6.5 seconds.  
10 Although not illustrated by the timeline of Fig. 7, if the remaining video content of  
11 Video11.asf 806 were not fully buffered by the end of the 6 seconds designated by  
12 the duration command 808, the ShowWhileBuffering command 810 would ensure  
13 that the FBIWarning.jpg still image 804 continued to be displayed until the  
14 remaining video content of Video11.asf 806 was fully buffered.

15 Various additional and alternative implementations are possible regarding  
16 the embodiment described above with respect to Figs. 6 - 8. As described above,  
17 the embodiment applies to video content that happens to begin with a still image  
18 (e.g., the FBIWarning.jpg). Thus, the still image 804 referenced in the stitched-  
19 reference play-list 208 is typically the first video frame of the video content  
20 extracted out by a content provider as a JPEG (Joint Photographic Experts Group),  
21 GIF (Graphics Interchange Format), PNG (Portable Network Graphics), or other  
22 similarly formatted image file format. However, in another alternate  
23 implementation, video content may not begin with a still image. In such an  
24 implementation, a content provider or redistributor may insert its own still image  
25 in JPEG, GIF, or PNG format. Such a still image may include a company logo

1 image, an advertisement image, and so on. Fig. 9 provides an example of such a  
2 still image wherein a redistributor has inserted a still image of its logo 900 to be  
3 displayed upon selection of video content so that the logo appears for a period of  
4 time as the selected video content buffers in the background.

5 In another alternate implementation, instead of a single still image being  
6 displayed, several still images might be manipulated and displayed to provide the  
7 effect of animation. Thus, instead of a stitched-reference play-list 208 referencing  
8 a single still image, it would reference several still images that would be buffered  
9 and displayed as an animated image prior to and during the buffering of the  
10 selected video content. Again, an animated image might include a company logo,  
11 an advertisement, and so on. Other implementations may provide variations which  
12 apply other digital animation effects to a still image. Such variations include, for  
13 example, slowly fading a still image in over a period of time while the video clip is  
14 buffering. The still image could also fade out or “crumble” out once the video clip  
15 is buffered and ready to play. Such digital animation effects can be specified as  
16 part of the play-list 208 format. As an example, tags can be included in an ASX  
17 play-list file that instruct the media player 200 to fade out the still image or  
18 produce a slow shattering effect of the still image when the video clip is ready to  
19 play. In general, such effects may be applicable in various alternate scenarios  
20 described throughout this disclosure, some of which are discussed below.

21 The stitched-reference play-list 208 of Fig. 10 illustrates another exemplary  
22 embodiment that achieves seamless startup and transitions between streaming  
23 video content on a streaming media player 200 using still images as interstitials. In  
24 this embodiment, a content provider can insert a still image in between video clips  
25 that are intended to play one after the other. Inserting the still image can smooth

1 over the transition between two video streams. Although a second video stream  
2 typically can be buffered as the last part of a first video stream is being played on a  
3 media player 200, the transition from playing the first video stream to playing the  
4 second video stream is often not a seamless transition. This may be because of  
5 network glitches or because the streaming for the first clip does not end in time to  
6 fully buffer the second clip. In such circumstances, a stitched-reference play-list  
7 208 such as that shown in Fig. 10 can be used to provide a seamless transition  
8 between video clips.

9 The stitched-reference play-list 208 of Fig. 10 refers to video content that  
10 combines 2 video clips, Video1.asf 1000 and Video2.asf 1002, for playback on a  
11 streaming media player 200. The media player 200 initially buffers and plays  
12 Video1.asf 1000. When playback of Video1.asf 1000 is coming to an end and  
13 Video1.asf 1000 is no longer buffering, a still image Logo.jpg 1004 buffers and is  
14 displayed by the streaming media player 200. As in the above embodiments, the  
15 still image 1004 takes only a very short time to buffer, causing it to be displayed  
16 virtually immediately upon the conclusion of Video1.asf 1000. Once the still  
17 image Logo.jpg 1004 begins being displayed, the media player 200 begins  
18 buffering Video2.asf 1002. The ShowWhileBuffering command 1006 following  
19 the reference to the still image Logo.jpg 1004 ensures that the still image Logo.jpg  
20 1004 will be displayed until the second video clip (i.e., Video2.asf 1002) is fully  
21 buffered and able to begin playback. In an alternate implementation, where  
22 available network bandwidth and other factors enable buffering of both the still  
23 image Logo.jpg 1004 and the second video clip, Video2.asf 1002, prior to the end  
24 of the currently playing Video1.asf 1000, the ShowWhileBuffering command 1006  
25 which follows the reference to the still image Logo.jpg 1004 enables the second

1 video clip, Video2.asf 1002, to begin playing directly after the first video clip,  
2 Video1.asf 1000, is completed. Thus, the still image Logo.jpg 1004 would not  
3 need to be displayed at all. The still image Logo.jpg 1004 could be skipped over  
4 and the media player could immediately begin playing the second video clip,  
5 Video2.asf 1002.

6 Also, as noted above, alternate implementations of this embodiment may  
7 include the use of several still images being manipulated and displayed to provide  
8 the effect of animation. Thus, instead of a stitched-reference play-list 208  
9 referencing a single still image, it may reference several still images that would be  
10 buffered and displayed as an animated image during the buffering of the second  
11 video clip. Again, an animated image might include a company logo, an  
12 advertisement, and so on. In addition, other implementations may provide  
13 variations that apply other digital animation effects to a still image. Such  
14 variations include, for example, slowly fading a still image in over a period of time  
15 while the second video clip is buffering. The still image could also fade out or  
16 “crumble” out once the second video clip is buffered and ready to play.

17 A variation of the stitched-reference play-list 208 embodied in Fig. 10 is  
18 shown in the play-list 208 of Fig. 11. The play-list 208 of Fig. 11 includes a  
19 “duration” command 1100 following the reference to the still image Logo.jpg 1004  
20 and preceding the ShowWhileBuffering command 1006. As discussed herein  
21 above with regard to another embodiment, this combination of duration and  
22 ShowWhileBuffering commands causes the streaming media player 200 to ensure  
23 that the still image Logo.jpg 1004 is displayed for at least the amount of time  
24 specified by the duration command 1008 (i.e., 6 seconds as shown in Fig. 11). If  
25 however, the second video clip, Video2.asf 1002, is not fully buffered by the time

1 the 6 seconds is completed, then the ShowWhileBuffering command 1006 will  
2 cause the media player 200 to continue displaying the still image Logo.jpg 1004  
3 until the video clip, Video2.asf 1002, is fully buffered. Again, as in the Fig. 10  
4 embodiment, variations of the Fig. 11 embodiment are possible, such as the use of  
5 several still images being manipulated to provide an animation, and digital  
6 animation effects applied that slowly fade a still image in or out over a period of  
7 time while the second video clip is buffering.

8 The stitched-reference play-list 208 of Fig. 12 illustrates still another  
9 exemplary embodiment that achieves seamless startup and transitions between  
10 streaming video content on a streaming media player 200 using still images as  
11 interstitials. In the Fig. 12 embodiment, the still image is not inserted by a content  
12 provider. Rather, the still image is the last frame of a preceding video clip. A  
13 content provider can ensure that the transition between a first video clip and a  
14 second video clip will not result in a blank screen in the event that the second  
15 video clip is not fully buffered by the time the first video clip has finished playing.  
16 This is accomplished by maintaining the streaming media player 200 display with  
17 the last frame of the first video clip until the second video clip is fully buffered. In  
18 another alternate implementation, several frames of the preceding video clip may  
19 be manipulated in such a way as to create an animated image when they are played  
20 back. As in the several embodiments discussed above, the Fig. 12 embodiment  
21 contemplates implementations in which digital animation effects can be applied to  
22 a single still image (e.g., the last frame of the preceding video clip), such as slowly  
23 fading a still image in or out over a period of time while the second video clip is  
24 buffering.

1 In a manner similar to the embodiments of Figs. 10 and 11, the stitched-  
2 reference play-list 208 of Fig. 12 refers to video content that combines 2 video  
3 clips, Video1.asf 1200 and Video2.asf 1202, for playback on a streaming media  
4 player 200. The media player 200 initially buffers and plays Video1.asf 1200.  
5 When playback of Video1.asf 1200 is coming to an end and Video1.asf 1200 is no  
6 longer buffering, Video2.asf 1202 begins to buffer. Because it is likely that  
7 Video1.asf 1200 and Video2.asf 1202 have the same streaming rate, and the steady  
8 state rate of streaming media content is typically the rate at which the streaming  
9 media content is encoded for playback, Video2.asf 1202 will likely begin playback  
10 directly upon completion of playback of Video1.asf 1200. That is, it is generally  
11 expected that Video2.asf 1202 will be fully buffered and ready to begin playback at  
12 precisely the time that playback of Video1.asf 1200 ends. However, due to various  
13 factors that may affect the data streaming rate, Video2.asf 1202 may not be fully  
14 buffered by the time playback of Video1.asf 1200 ends. In the past, this scenario  
15 would result in a blank media screen between the time when playback of  
16 Video1.asf 1200 ended and the time when Video2.asf 1202 was fully buffered.  
17 However, the ShowWhileBuffering command 1204 included in the stitched-  
18 reference play-list 208 of Fig. 12 functions to prevent a blank media screen from  
19 being displayed under such circumstances. The ShowWhileBuffering command  
20 1204 causes a streaming media player 200 to maintain the display with a still  
21 image of the last frame of Video1.asf 1200 until such time as Video2.asf 1202 is  
22 fully buffered. Thus, there is a seamless and smooth transition between the two  
23 pieces of streaming video content.

24 It is noted that alternate embodiments are contemplated that include various  
25 combinations and/or extensions of the embodiments illustrated by the stitched

1 reference play-lists 208 of Figs. 8, 10, 11 and 12. For example, the embodiment of  
2 Fig. 8 generally illustrates the use of a still image 804 to achieve quick startup for  
3 video content and certain play-list commands (i.e., Duration and  
4 ShowWhileBuffering commands) to control the length of display for the still  
5 image 804 until such time as video content (i.e., Video 11.asf 806) is buffered and  
6 ready for playback. The embodiment of Fig. 10 generally illustrates a quick  
7 startup and smooth transition between two video clips. Thus, in an alternate  
8 embodiment, a play-list 208 such as that shown in Fig. 8 may be combined  
9 appropriately with a play-list 208 such as that shown in Fig. 10 to provide quick  
10 startup for video content and smooth transitions between video clips with respect  
11 to any number of video clips that may be stitched together in a stitched-reference  
12 play-list 208.

### 13 14 **Exemplary Methods**

15 Example methods for achieving quick startup of video content in a  
16 streaming media player will now be described with primary reference to the flow  
17 diagrams of Figs. 13 - 17. The methods apply generally to the exemplary  
18 embodiments discussed above with respect to Fig. 2 and Figs. 6 - 12. The  
19 elements of the described methods may be performed by any appropriate means  
20 including, for example, by hardware logic blocks on an ASIC or by the execution  
21 of processor-readable instructions defined on a processor-readable medium.

22 A "processor-readable medium," as used herein, can be any means that can  
23 contain, store, communicate, propagate, or transport instructions for use by or  
24 execution by a processor. A processor-readable medium can be, without  
25 limitation, an electronic, magnetic, optical, electromagnetic, infrared, or

1 semiconductor system, apparatus, device, or propagation medium. More specific  
2 examples of a processor-readable medium include, among others, an electrical  
3 connection (electronic) having one or more wires, a portable computer diskette  
4 (magnetic), a random access memory (RAM) (magnetic), a read-only memory  
5 (ROM) (magnetic), an erasable programmable-read-only memory (EPROM or  
6 Flash memory), an optical fiber (optical), a rewritable compact disc (CD-RW)  
7 (optical), and a portable compact disc read-only memory (CDROM) (optical).

8 Fig. 13 shows an exemplary method 1300 for achieving quick startup of  
9 video content in a streaming media player 200. At block 1302, a streaming media  
10 player 200 (e.g., on a client computing device 102) receives a static image from a  
11 content provider, such as a content server 104. Typically, the receipt of the static  
12 image is the result of a user making a video content selection that directs the  
13 streaming media player 200 to a play-list, such as a stitched-reference play-list  
14 208. A stitched-reference play-list 208 refers the streaming media player 200 to  
15 the static image which the media player 200 requests from the content provider.  
16 The static image promptly downloads (e.g., within approximately 0.5 seconds) to  
17 the media player 200 where it can be displayed.

18 At block 1304 of method 1300, the streaming media player 200 displays the  
19 static image on a monitor or other type of display device associated with client  
20 computing device 102. While the static image is displayed, the streaming media  
21 player 200 buffers video content from the content provider, as shown at block  
22 1306. The video content is referenced in the stitched-reference play-list 208 being  
23 implemented by the streaming media player 200. At block 1308, when the  
24 buffering of the video content is complete, the streaming media player 200 stops  
25 displaying the static image and begins playing back the video content. Thus, the

1 static image is displayed quickly and for as long as it takes the video content to  
2 buffer and be ready for playback.

3 Fig. 14 shows another exemplary method 1400 for achieving quick startup  
4 of video content in a streaming media player 200. At block 1402, a streaming  
5 media player 200 (e.g., on a client computing device 102) receives a static image  
6 from a content provider, such as a content server 104. As in the previous method,  
7 the receipt of the static image is typically the result of a user making a video  
8 content selection that directs the streaming media player 200 to a stitched-  
9 reference play-list 208. The static image quickly downloads to the media player  
10 200 where it can be displayed.

11 At block 1404, the streaming media player 200 displays the static image for  
12 a fixed duration on a monitor or other display device associated with client  
13 computing device 102. The fixed duration is an amount of time (e.g., several  
14 seconds) specified by a “duration” command in the stitched-reference play-list  
15 208. While the static image is displayed, the streaming media player 200 buffers  
16 video content from the content provider, as shown at block 1406. If the fixed  
17 duration expires and the video content is not fully buffered, the static image can be  
18 displayed beyond the fixed duration, as shown at block 1408. Thereafter, once the  
19 buffering of the video content is complete, the streaming media player 200 stops  
20 displaying the static image and begins playing back the video content as shown at  
21 block 1410. Thus, the static image is displayed for at least the duration specified  
22 in the duration command and maybe longer if the video content requires additional  
23 time to be fully buffered.

24 At block 1412, another alternative is that the buffering of the video content  
25 is complete when the fixed duration for displaying the static image expires. In this

1 case, the streaming media player 200 would stop displaying the static image and  
2 begin playing back the video content.

3 Fig. 15 shows another exemplary method 1500 for achieving quick startup  
4 and a smooth transition between two video clips in a streaming media player 200.  
5 At block 1502, a streaming media player 200 plays a first video clip. While  
6 playing the first video clip, the streaming media player 200 begins buffering a  
7 second video clip as shown in block 1504. Typically, the second video clip would  
8 be expected to be buffered just in time to begin playback when the first video clip  
9 completed playback. However, due to various factors that may affect the data  
10 streaming rate, the second video clip may not be fully buffered by the time  
11 playback of the first video clip ends. Thus, at block 1506, the streaming media  
12 player 200 displays the last frame of the first video clip. The last frame of the first  
13 video clip continues to be displayed until the second video clip is fully buffered.  
14 As indicated at block 1508, when the buffering of the second video clip is  
15 completed, the streaming media player 200 stops displaying the last frame of the  
16 first video clip and begins playing back the second video clip.

17 Fig. 16 shows another exemplary method 1600 for achieving quick startup  
18 and a smooth transition between two video clips in a streaming media player 200.  
19 At block 1602, a streaming media player 200 plays a first video clip. While  
20 playing the first video clip, the streaming media player 200 buffers a static image  
21 as shown at block 1604. At block 1606, the static image is displayed upon the  
22 completion of playing the first video clip. While the static image is being  
23 displayed, a second video clip is buffered as shown at block 1608. At block 1610,  
24 the second video clip begins playing back (and the static image is no longer  
25 displayed) when the buffering for the second video clip is complete.

1        Fig. 17 shows another exemplary method 1700 for achieving quick startup  
2 and a smooth transition between two video clips in a streaming media player 200.  
3 At block 1702, a streaming media player 200 plays a first video clip. While  
4 playing the first video clip, the streaming media player 200 buffers a static image  
5 as shown at block 1704. At block 1706, the static image is displayed for a fixed  
6 duration upon the completion of playing the first video clip. The fixed duration is  
7 an amount of time (e.g., several seconds) specified by a “duration” command in a  
8 stitched-reference play-list 208 being implemented by the streaming media player  
9 200. While the static image is being displayed, a second video clip is buffered as  
10 shown at block 1708.

11        If the fixed duration expires and the second video clip is not fully buffered,  
12 the static image can be displayed beyond the fixed duration, as shown at block  
13 1710. Thereafter, once the buffering of the second video clip is complete, the  
14 streaming media player 200 stops displaying the static image and begins playing  
15 back the second video clip as shown at block 1712. Thus, the static image is  
16 displayed for at least the duration specified in the duration command and maybe  
17 longer if the second video clip requires additional time to be fully buffered.

18        At block 1714, another alternative is that the buffering of the second video  
19 clip is complete when the fixed duration for displaying the static image expires. In  
20 this case, the streaming media player 200 would stop displaying the static image  
21 and begin playing back the second video clip.

22        While one or more methods have been disclosed by means of flow diagrams  
23 and text associated with the blocks of the flow diagrams, it is to be understood that  
24 the blocks do not necessarily have to be performed in the order in which they were  
25 presented, and that an alternative order may result in similar advantages.

1 Furthermore, the methods are not exclusive and can be performed alone or in  
2 combination with one another.

### 3 4 Exemplary Computer

5 Fig. 18 illustrates an exemplary computing environment suitable for  
6 implementing a client computing device 102 and a content server computing  
7 device 104. Although one specific configuration is shown, client computing  
8 device 102 and content server computing device 104 may be implemented in other  
9 computing configurations.

10 The computing environment 1800 includes a general-purpose computing  
11 system in the form of a computer 1802. The components of computer 1802 can  
12 include, but are not limited to, one or more processors or processing units 1804, a  
13 system memory 1806, and a system bus 1808 that couples various system  
14 components including the processor 1804 to the system memory 1806.

15 The system bus 1808 represents one or more of any of several types of bus  
16 structures, including a memory bus or memory controller, a peripheral bus, an  
17 accelerated graphics port, and a processor or local bus using any of a variety of bus  
18 architectures. An example of a system bus 1808 would be a Peripheral Component  
19 Interconnects (PCI) bus, also known as a Mezzanine bus.

20 Computer 1802 typically includes a variety of computer readable media.  
21 Such media can be any available media that is accessible by computer 1802 and  
22 includes both volatile and non-volatile media, removable and non-removable  
23 media. The system memory 1806 includes computer readable media in the form of  
24 volatile memory, such as random access memory (RAM) 1810, and/or non-volatile  
25 memory, such as read only memory (ROM) 1812. A basic input/output system

1 (BIOS) 1814, containing the basic routines that help to transfer information  
2 between elements within computer 1802, such as during start-up, is stored in ROM  
3 1812. RAM 1810 typically contains data and/or program modules that are  
4 immediately accessible to and/or presently operated on by the processing unit  
5 1804.

6 Computer 1802 can also include other removable/non-removable,  
7 volatile/non-volatile computer storage media. By way of example, Fig. 18  
8 illustrates a hard disk drive 1816 for reading from and writing to a non-removable,  
9 non-volatile magnetic media (not shown), a magnetic disk drive 1818 for reading  
10 from and writing to a removable, non-volatile magnetic disk 1820 (e.g., a “floppy  
11 disk”), and an optical disk drive 1822 for reading from and/or writing to a  
12 removable, non-volatile optical disk 1824 such as a CD-ROM, DVD-ROM, or  
13 other optical media. The hard disk drive 1816, magnetic disk drive 1818, and  
14 optical disk drive 1822 are each connected to the system bus 1808 by one or more  
15 data media interfaces 1826. Alternatively, the hard disk drive 1816, magnetic disk  
16 drive 1818, and optical disk drive 1822 can be connected to the system bus 1808  
17 by a SCSI interface (not shown).

18 The disk drives and their associated computer-readable media provide non-  
19 volatile storage of computer readable instructions, data structures, program  
20 modules, and other data for computer 1802. Although the example illustrates a  
21 hard disk 1816, a removable magnetic disk 1820, and a removable optical disk  
22 1824, it is to be appreciated that other types of computer readable media which can  
23 store data that is accessible by a computer, such as magnetic cassettes or other  
24 magnetic storage devices, flash memory cards, CD-ROM, digital versatile disks  
25 (DVD) or other optical storage, random access memories (RAM), read only

1 memories (ROM), electrically erasable programmable read-only memory  
2 (EEPROM), and the like, can also be utilized to implement the exemplary  
3 computing system and environment.

4 Any number of program modules can be stored on the hard disk 1816,  
5 magnetic disk 1820, optical disk 1824, ROM 1812, and/or RAM 1810, including  
6 by way of example, an operating system 1826, one or more application programs  
7 1828, other program modules 1830, and program data 1832. Each of such  
8 operating system 1826, one or more application programs 1828, other program  
9 modules 1830, and program data 1832 (or some combination thereof) may include  
10 an embodiment of a caching scheme for user network access information.

11 Computer 1802 can include a variety of computer/processor readable media  
12 identified as communication media. Communication media typically embodies  
13 computer readable instructions, data structures, program modules, or other data in  
14 a modulated data signal such as a carrier wave or other transport mechanism and  
15 includes any information delivery media. The term “modulated data signal” means  
16 a signal that has one or more of its characteristics set or changed in such a manner  
17 as to encode information in the signal. By way of example, and not limitation,  
18 communication media includes wired media such as a wired network or direct-  
19 wired connection, and wireless media such as acoustic, RF, infrared, and other  
20 wireless media. Combinations of any of the above are also included within the  
21 scope of computer readable media.

22 A user can enter commands and information into computer system 1802 via  
23 input devices such as a keyboard 1834 and a pointing device 1836 (e.g., a  
24 “mouse”). Other input devices 1838 (not shown specifically) may include a  
25 microphone, joystick, game pad, satellite dish, serial port, scanner, and/or the like.

1 These and other input devices are connected to the processing unit 1804 via  
2 input/output interfaces 1840 that are coupled to the system bus 1808, but may be  
3 connected by other interface and bus structures, such as a parallel port, game port,  
4 or a universal serial bus (USB).

5 A monitor 1842 or other type of display device can also be connected to the  
6 system bus 1808 via an interface, such as a video adapter 1844. In addition to the  
7 monitor 1842, other output peripheral devices can include components such as  
8 speakers (not shown) and a printer 1846 which can be connected to computer 1802  
9 via the input/output interfaces 1840.

10 Computer 1802 can operate in a networked environment using logical  
11 connections to one or more remote computers, such as a remote computing device  
12 1848. By way of example, the remote computing device 1848 can be a personal  
13 computer, portable computer, a server, a router, a network computer, a peer device  
14 or other common network node, and the like. The remote computing device 1848  
15 is illustrated as a portable computer that can include many or all of the elements  
16 and features described herein relative to computer system 1802.

17 Logical connections between computer 1802 and the remote computer 1848  
18 are depicted as a local area network (LAN) 1850 and a general wide area network  
19 (WAN) 1852. Such networking environments are commonplace in offices,  
20 enterprise-wide computer networks, intranets, and the Internet. When  
21 implemented in a LAN networking environment, the computer 1802 is connected  
22 to a local network 1850 via a network interface or adapter 1854. When  
23 implemented in a WAN networking environment, the computer 1802 typically  
24 includes a modem 1856 or other means for establishing communications over the  
25 wide network 1852. The modem 1856, which can be internal or external to

1 computer 1802, can be connected to the system bus 1808 via the input/output  
2 interfaces 1840 or other appropriate mechanisms. It is to be appreciated that the  
3 illustrated network connections are exemplary and that other means of establishing  
4 communication link(s) between the computers 1802 and 1848 can be employed.

5 In a networked environment, such as that illustrated with computing  
6 environment 1800, program modules depicted relative to the computer 1802, or  
7 portions thereof, may be stored in a remote memory storage device. By way of  
8 example, remote application programs 1858 reside on a memory device of remote  
9 computer 1848. For purposes of illustration, application programs and other  
10 executable program components, such as the operating system, are illustrated  
11 herein as discrete blocks, although it is recognized that such programs and  
12 components reside at various times in different storage components of the  
13 computer system 1802, and are executed by the data processor(s) of the computer.

### 14 15 **Conclusion**

16 Although the invention has been described in language specific to structural  
17 features and/or methodological acts, it is to be understood that the invention  
18 defined in the appended claims is not necessarily limited to the specific features or  
19 acts described. Rather, the specific features and acts are disclosed as exemplary  
20 forms of implementing the claimed invention.